GRANT/AMES

16 pages

IN-28443

A View of Kanerva's Sparse Distributed Memory

Peter J. Denning

June 4, 1986

Research Institute for Advanced Computer Science
NASA Ames Research Center

RIACS Technical Report 86.14

(NASA-TM-89206) A VIEW OF KANERVA'S SPARSE DISTRIBUTED MEMORY (NASA) 16 p CSCL 09B

N86-33030

Unclas
G3/60 44588

RIACS

Research Institute for Advanced Computer Science

A View of Kanerva's Sparse Distributed Memory

Peter J. Denning

Research Institute for Advanced Computer Science
NASA Ames Research Center

RIACS Technical Report 86.14
June 4, 1986

Pentti Kanerva of RIACS has been working on a machine of a new class of computers, which he calls pattern computers. Pattern computers may close the gap between capabilities of biological organisms to recognize and act on patterns -- visual, auditory, tactile, or olfactory -- and capabilities of modern computers. Combinations of numeric, symbolic, and pattern computers may one day be capable of sustaining robots. This essay gives an overview of the requirements for a pattern computer, a summary of Kanerva's Sparse Distributed Memory (SDM), and examples of tasks this computer can be expected to perform well.

Work reported herein was supported in part by Cooperative Agreement NCC 2-387 between the National Aeronautics and Space Administration (NASA) and the Universities Space Research Association (USRA).

A View of Kanerva's Sparse Distributed Memory

Peter J. Denning

Research Institute for Advanced Computer Science

June 4, 1986

Today's computers excel at two kinds of task. Numeric computers far exceed human capacity in performing complex calculations such as in solving equations of physical systems or supporting business operations. Symbolic computers far exceed human capacity in processing strings of characters such as in logical deduction and text processing. Neither of these types of computers has come close to biological organisms in recognizing patterns or retrieving stored patterns -- visual, auditory, tactile, or olfactory. Humans can, in less than one-tenth of a second, recognize a familiar face in a crowd. They can effortlessly recognize English spoken by a wide variety of persons with many different voices. They can recognize many variations on the letter "A," or a musical theme, or any other pattern. No computer has come close to duplicating these feats.

Pentti Kanerva of RIACS has been working on a machine of a new class of computers, which he calls pattern computers, that may close this gap. In this essay I will give an overview of the requirements on a pattern computer, a summary of the architecture Kanerva calls Sparse Distributed Memory (SDM), and

examples of tasks this computer can be expected to perform well. Combinations of numeric, symbolic, and pattern computers may one day be capable of sustaining robots.

In *Mind over Machine*, Hubert and Stuart Dreyfus discuss five levels of human skill: novice, advanced beginner, competent performer, proficient performer, and expert (*1*). The lowest level is characterized by analysis of situations and applications of basic rules to calculate successful action. The highest level is characterized by recall of abstractions of similar past situations, the memories of which contain past actions. The lowest level uses a slow, conscious process of deduction and rule manipulation; the highest level uses a fast, unconscious lookup of a pattern containing suggested actions.

The Dreyfuses are troubled by the failure of AI research to build computers that can reproduce human skills faithfully. They argue that contemporary computers, which are either numeric or symbolic, are well suited to rule manipulation and searches characteristic of low skill levels; but because they cannot perform fast pattern recalls, or quickly form abstractions of sets of similar past patterns, these computers cannot move much beyond the stage of bare competence. The Dreyfuses speculate that computers with mathematical properties like holograms are suited for such tasks: two holograms can be quickly checked for similarities (by shining light through both); loss of information in a local region does not destroy the set of patterns retrievable.

In *Brains, Behavior, and Robotics*, James Albus argues that many of the functions of organisms arise from their structure (*2*). He describes a Cerebellar Model Arithmetic Computer (CMAC), whose internal structure reproduces key functions of the human nervous system. He argues that this type of computer is capable of processing large patterns efficiently and is likely to lead to good robots.

A growing number of researchers share the belief that neural networks, which are systems of interacting threshold logic elements, may make good models for the pattern processing properties of people. These networks can store large binary patterns as their stable states. Although neural networks were first proposed by McCulloch and Pitts in 1943, their mathematical richness was not appreciated until recently. One of the most important models of this type was proposed by John Hopfield in 1982 (*3*). Collections of papers about other neural-net architectures have been edited by Hinton and Anderson (*4*) and by Rumelhart and McClelland (*5, 6*).

In analyzing descriptions of human expert skills. discussions of the nervous system. and properties of neural networks. one can deduce that the requirements for pattern computation include:

1. Able to look up very large patterns (10^5 bits or more).
2. Able to cycle 10-1000 times per second.
3. Able to link patterns and to recall pattern sequences.

4. Able to look up patterns similar to a given pattern.

5. Able to generate a pattern that is an abstraction of a given set of similar patterns.

6. Able to continue functioning, perhaps degraded, if a local portion of the storage system fails.

Ordinary random-access memories are optimized for very small patterns: for example, 32-bit words with cycle times on the order of 10^{-8} second. They meet only the second and third requirements. Associative (content-addressable) memories are designed to find locations whose contents exactly match the subset of address bits determined by a mask. They can meet the first, second, and third requirements. An attempt to meet the fourth requirement by providing a set of masks enumerating all similarities would lead to a hopelessly complex memory structure when the number of address bits becomes large. Memories of new architecture are required to meet all the requirements.

Figure 1 shows Kanerva's model of a pattern computer. It is inspired by mathematical models of human memory (*7, 8*). The *Focus* is a processing element that receives a code representing current sensory input and a pattern from the memory; it produces a new pattern for the memory and a code that drives motor apparatus. The Focus also contains a short-term memory, which holds a small number of recent memory patterns (e.g., 7). The internal state of the Focus is a code for the current moment of experience. The Focus can search the memory by cycling through a sequence of patterns.

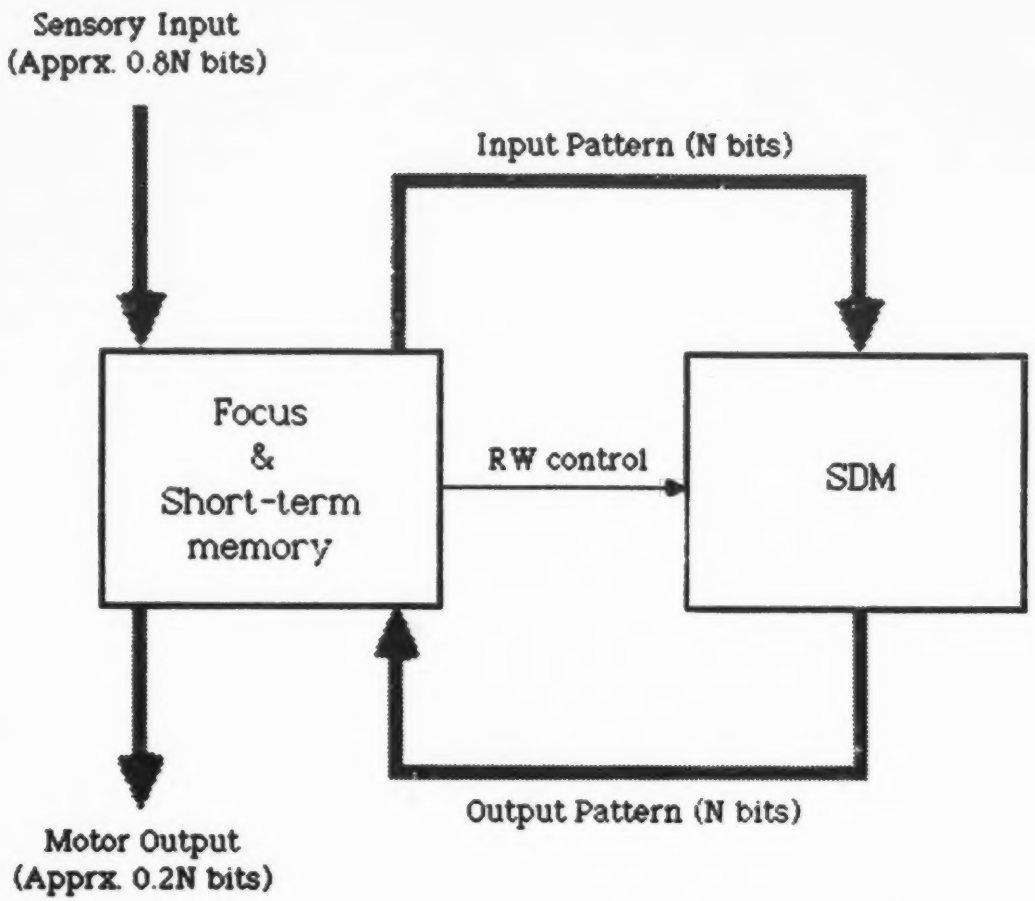

FIGURE 1. Pattern computer based on SDM.

The number of pattern bits is denoted by N. (Kanerva believes that the number of sensory bits is about 80% of N and the number of motor bits about 20%.) For human memory, N may be on the order of 10^6 bits or more and the cycle time of the Focus on the order of 0.1 second. A computer of this form may be able to trade pattern width for cycle time -- for example, N on the order of 10^4 bits and cycle time 0.001 second. A memory that meets the requirements is described below. (See Figure 2.)

The address space consists of 2^N potential locations. The set of M actual locations, called cells, are assigned N-bit addresses at random. A cell is selected by the memory's input pattern if its address is within Hamming distance D bits of that input. Equivalently, the input pattern is a point in an N-dimensional space: all cells within a (hyper)sphere of radius D are selected by that input. (Kanerva recommends choosing D so that 0.05 to 1.0 per cent of the cells are selected.) Each cell contains N counters. A write access to the memory stores the data pattern into each selected cell by adding one to each counter corresponding to a 1-bit of the data, and subtracting one from each counter corresponding to a 0-bit of the data. A read access retrieves data by reconstructing it from the sphere of selected cells using a majority voting rule: if the sum of all counters in a particular bit position is positive, output a 1-bit in that position, otherwise output a 0-bit. Kanerva has shown that 8 bits (one standard byte) is sufficient for each counter. He has also shown that all the operations -- selecting cells, adjusting counters, and reading counters -- can be done with

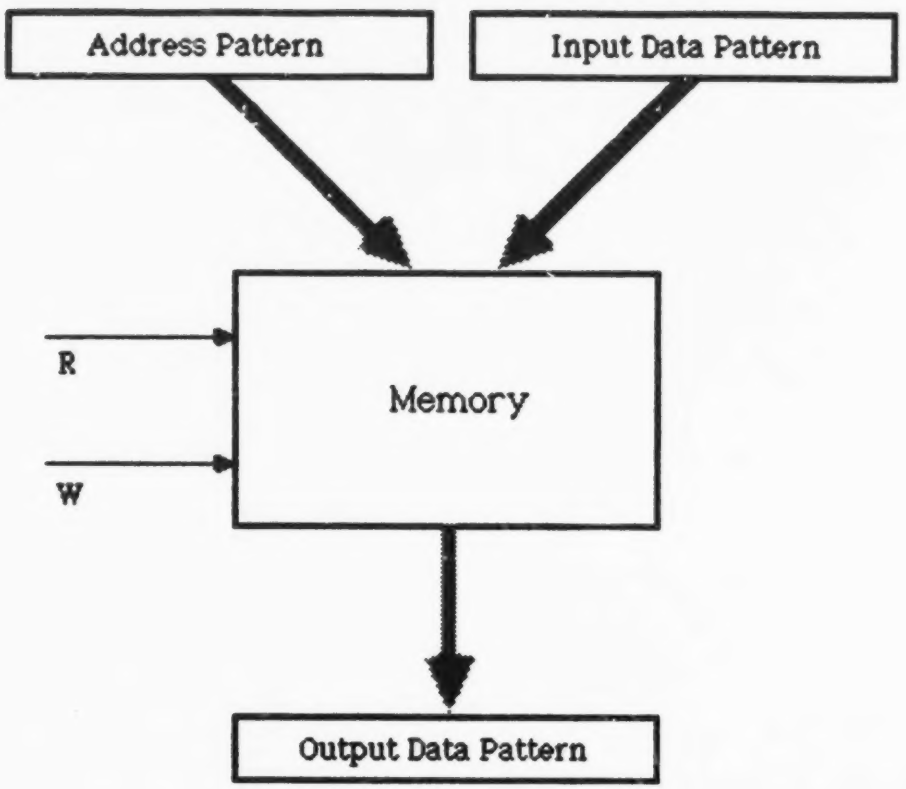

FIGURE 2. Memory interface.

linear threshold logic.

It is now apparent where the Sparse Distributed Memory gets its name. The set of M cells is sparsely embedded in the address space of 2^N potential locations; each stored pattern is distributed over a set of cells. Because each pattern is distributed over many cells, the number of patterns that can be stored is less than the number of cells. Kanerva shows that the theoretical capacity of the SDM is about $M/10$ patterns and its useful capacity is about $M/100$ patterns.

How does the SDM meet each of the requirements for a pattern computer? Kanerva argues as follows.

1. By design, the memory handles large patterns.
2. As shown in Table 1, cycle times for sample simulated SDMs for various choices of the parameters (N, M, D) are within the required range.
3. The SDM is a generalization of random-access memory for large patterns. A link between patterns A and B can be established by storing B in the sphere centered on A.
4. By design, the memory can retrieve patterns similar to the address pattern. Stored patterns within a critical distance of the input pattern can be retrieved. (The critical distance is about $D/2$ when the number of patterns stored is less than $M/100$, and decreases to 0 as the number of stored patterns increases to $M/10$.)

Table 1: Examples of SDM

Hardware	Dimension *N*	Cells *M*	Sphere *D*	Cycles per second
Dedicated DEC 2060	128	10,000	46-51	0.2-1.0
32-node Intel iPSC hypercube	128	50,000	46-51	1-5
Breadboard prototype	256	100,000	103-107	10-100
64,000-node Connection Machine	256	180,000	103-107	50-200
Largest feasible prototype (present VLSI)	1,000	100,000,000	448-459	1,000

Note: The breadboard prototype would cost about $75,000 for about half the performance of the $3,000,000 Connection Machine.

5. It is a research problem how to find codes for sensory input and motor actions that allow similar patterns to be stored in the same regions of the address space. If this can be done, a pattern that addresses the center of the overlapping regions would become an abstraction for the set of regional patterns. It appears that such codes can be found in simple cases.

6. Because many cells participate in the storage of one pattern, the memory will continue to function, perhaps degraded, if a local region should fail or be obliterated by repetitive storage of other patterns.

To reveal why the SDM computer is capable of pattern processing on an order that may support skilled behavior, it is helpful to consider a series of

increasingly complicated examples. These examples are inspired by the speech-processing application that will serve as one of the first tests of the SDM computer when the simulators are working by late 1986.

Consider a restricted version of the computer in Figure 1 to be used for speech recognition. The sensory input is derived from audio equipment through preprocessors that create one auditory code for each spoken word. (This assumption, one code per word rather than one code per phoneme, is not important and will be removed later.) The output (motor) codes are ASCII strings corresponding to the spoken words in the auditory input.

To train the memory, we need simply to speak selected words, thereby generating their auditory codes, and then write the corresponding ASCII code in the memory sphere selected by each given auditory code. Thus if someone speaks the phrase, "Mary had a little lamb," the training process will yield five spheres containing the ASCII codes for the words. Now the memory can be switched to retrieve mode. When any of the words is spoken, the memory will retrieve its ASCII code and make it available as output.

A simple extension of the above coding scheme will allow the memory to retain the fact that a sequence of words has meaning as a unit. In the sphere selected by a given word, we store that word's ASCII code and the auditory code of the *next* spoken word in the sequence. (This encoding scheme uses the short-term memory in the Focus.) The result of speaking "Mary had a little lamb" will be a linked chain of spheres in the SDM. Now the single spoken word "Mary"

will retrieve the first ASCII code plus the link for the head of the rest of the chain, thereby allowing successive cycles of the Focus to retrieve the rest of the sequence without further auditory input. In fact, speaking any word in a sequence would initiate the retrieval of the remaining sequence.

What would happen if the training processes described above were performed when the memory is not initially empty? As above, speaking a sequence of words distributes the codes over a chain of spheres; but now the strength of these codes may be too weak relative to other codes also distributed among the same spheres, and retrieval is impossible. The phrase must be repeated several times so that its relative strength in these spheres rises and retrieval becomes possible.

Actually the spoken repetitions are not identical. The auditory codes from successive repetitions of the same word will differ slightly according to the tone of voice, amount of stress, pitch, timber, sex, room accoustics, and background noises surrounding the speaker. Each repetition's code will address a slightly different sphere; but as long as the set of auditory codes for the same word have most of the bits the same -- within radius $D/2$ -- the set of spheres will overlap strongly. Thus the chain of spheres for each repetition will overlap strongly with the chains from prior repetitions. Because all the spheres for a given word will contain the same ASCII code, each spoken variant will retrieve its ASCII code properly. Moreover, new variants are highly likely to retrieve the same ASCII codes when the SDM "hears" them for the first time.

What happens if a new phrase is spoken that overlaps with a previous one? Consider "Mary, Mary quite contrary." The first chain overlaps strongly with the head(s) of the chain(s) for "Mary had a little lamb." After enough repetitions, there will be two sets of chains, one for each phrase. On hearing the word "Mary," which chain the SDM would retrieve would depend on the secondary bits of the auditory code (details of voice, noise, etc.), on the relative strengths of the two sets of chains, and on other information distributed among the same cells. Likewise, more than the immediately preceding word is needed to get past the second "Mary," but in general once a chain is picked, it can be followed easily to its conclusion.

These descriptions illustrate some aspects of human memory that can be simulated naturally by the SDM:

1. Easy association of outputs with sensory input patterns.
2. Increased strength of memory after more repetitions.
3. Storing items in sequence corresponding to their occurrence in time, and retrieving the tail of the sequence given the sensory input corresponding to any member.
4. Retrieving proper outputs despite variations of input.
5. Triggering retrieval of overlapping chains based on secondary bits (noise) in the sensory input.

The discussion above is meant to suggest the capabilities of a pattern computer based on SDM. To accomplish the tasks of speech recognition and document retrieval outlined above, much work remains. For example, the real-time output of audio equipment is likely to be in the form of phonemes rather than word-codes. We need to modify the encoding scheme in the SDM so that phoneme-sequences terminate on ASCII codes for blocks of letters. A second example is the encoding of links into stored pattern-sequences. The most general approach is to use the entire contents of the short-term memory (in the Focus) to link a pattern with a small number of preceding and following patterns. A third example is more ambitious than recognizing continuously-spoken speech: recognizing components of images.

References:

1. Hubert Dreyfus and Stuart Dreyfus. 1986. *Mind over Machine.* Macmillan, The Free Press. Excerpts appear in "Why computers may never think like people." *Technology Review.* January.

2. James Albus. 1981. *Brains, Behavior, and Robotics.* BYTE Books of McGraw-Hill.

3. John Hopfield. 1982. "Neural networks and physical systems with emergent collective computational abilities." *Proc. Nat. Acad. Sci. 79* (Biophysics), No. 8.

4. G. E. Hinton and J. A. Anderson (Eds.). 1981. *Parallel Models of Associative Memory.* Lawrence Erlbaum Associates.

5. D. E. Rumelhart and J. L. McClelland (Eds.). 1986. *Parallel Distributed Processing 1.* Bradford Books of MIT Press.

6. J. L. McClelland and D. E. Rumelhart (Eds.). 1986. *Parallel Distributed Processing 2.* Bradford Books of MIT Press.

7. Pentti Kanerva. 1986. "Parallel Structures in Human and Computer Memory." RIACS Technical Report TR-86.2. 6pp.

8. Pentti Kanerva. 1986 (est). *Self-propagating search: a unified theory of memory.* Bradford Books (MIT Press). In press.